ABSTRACT

Multiple parallel passive threads of instructions coordinate access to shared resources using "active" and "proactive" semaphores. The active semaphores send messages to execution and/or control circuitry to cause the state of a thread to change. A thread can be placed in an inactive state by a thread scheduler in response to an unresolved dependency, which can be indicated by a semaphore. A thread state variable corresponding to the dependency is used to indicate that the thread is in inactive mode. When the dependency is resolved a message is passed to control circuitry causing the dependency variable to be cleared. In response to the cleared dependency variable the thread is placed in an active state. Execution can proceed on the threads in the active state. A proactive semaphore operates in a similar manner except that the semaphore is configured by the thread dispatcher before or after the thread is dispatched to the execution circuitry for execution.